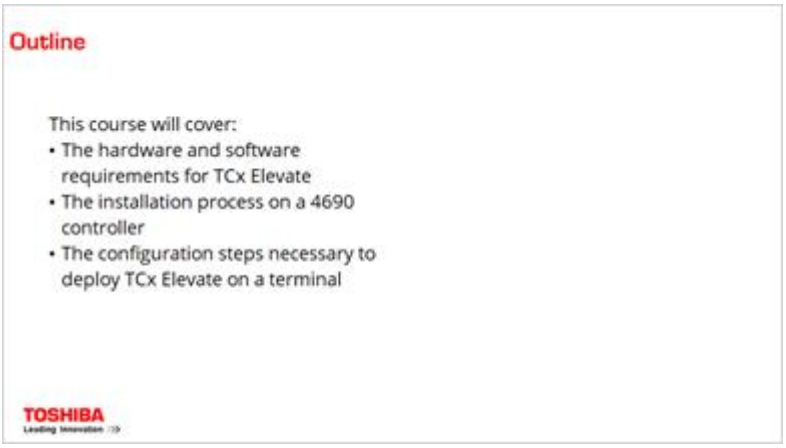
ELV 301 - Installing the TCx Elevate Platform on 4690

1.2 Title Slide

Hi, Welcome to ELV301: Installing the TCx Elevate platform on 4690.

1.3 Course Outline



In this course, we will cover:

- The hardware and software requirements for TCx Elevate
- The installation process on a 4690 controller
- and the configuration steps necessary to deploy TCx Elevate on a terminal.

1.4 Requirements Header

Let’s first take a look at the system requirements to install TCx Elevate.

1.5 Hardware Requirements

For the hardware requirements on the controller, you need a 4800 C45/C85 or a 4900 C46/C86 or above with at least one gigabyte of memory.

For a terminal with a single display, you'll need a 743 or newer with at least 2 gigabytes of memory. And for a terminal with a dual display, you’ll need a 745 or newer with at least 4 gigabytes of memory.

1.6 Software Requirements

Software requirements for TCx Elevate are as follows:

For the 4690 OS, you'll need V6R5 or newer. For Store Integrator, you'll need V4R1 or newer. If you're using ACE, you'll need at least V7R5. For SA, you'll need version S001 or newer, and for GSA, version Q001 or newer.

1.7 Installation Process Header

Now let's see the process for installing TCx Elevate on the controller.

1.8 Installation Video

To start the installation process, first insert the installation CD.

There are actually two CDs, one that installs the TCx Elevate platform with the sample apps, and one without the sample apps.

I'm installing the apps with the platform, but the process is the same either way.

With the CD in the master controller, switch to command mode, and enter the command P:\ccfinst to open the installation menu.

CCF stands for consumer client framework and refers to this portion of the larger TCx Elevate product. I only mention this because you will see those letters (CCF) pop-up from time to time throughout the installation and configuration process, and it's nice to know what they mean.

In the installation menu, you'll see three options: install, view logs and exit.

Enter 1 to begin the install.

Once the installation is complete, you can check the log for any errors.

If there are none, we can exit the installer and use ASM to apply the changes.

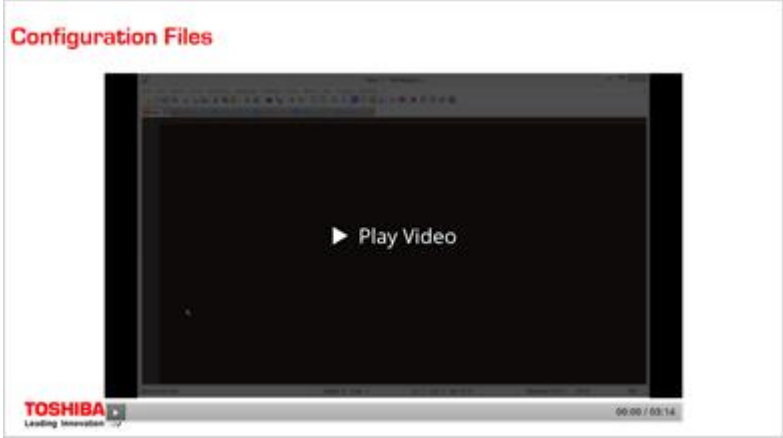
Here you can either put the changes into test mode or accept them. In a test environment you would want to use test mode first, and then come back and accept the changes later, but I am going to go ahead and accept the changes now.

Once ASM is complete, the installation process is technically complete, but there are still a few configuration changes that need to be made. We will address those in a separate video.

1.9 Terminal Config Header

After installation, there is some configuration that needs to be done in order to get TCx Elevate deployed on a terminal. We'll take a look at those steps now.

1.10 Configuration Files



After Elevate has been installed, we need to start the configuration process. Before we do that though, there are a few files that get copied to the controller to help us with config, and I want to talk about those.

The first is preload.ccf. In order to run elevate on a terminal, we need to make sure all the files that elevate needs are loaded to the terminals. This file defines a preload bundle that we will assign to our elevate terminals. The main thing to point out here is that the bundle includes everything in the M:\adxetc\ext\xpd folder of the controller. We will take a look at the contents of that folder a little later.

The next file is javaapp.ccf. This file defines a new java app to run on the terminal. You can see the app name is No Opp CCF. When we do our terminal config a little later, you'll see that we launch this "no op" application instead of our POS application. When using elevate, we actually launch the POS app through the elevate platform itself (which you'll see later) so we don't want to set the POS app to launch via the terminal config.

Also included with the install is a bat file, ccfgtc.bat which uses the 4690 config utility to update your existing preload bundle and javaapp definitions with those that are defined in preload.ccf and javaapp.ccf. So all we need to do is run this bat file on the controller to update those definitions.

The next file, adxcfgai.ccf defines two terminal applications. The applications defined in this file are what are called "Composite Applications". You can see CCFCOP for the operator display, and CCFCUST for the customer display. These are only the default composite applications and you will very likely want to create more than these two.

There are a few things I want to talk about with these definitions. If you look at the program arguments for the customer, you can see display 2. That means that this application is meant to run on the customer display. Notice there is not a display:1 for the operator application. That's just because display:1 is default if not specified.

You'll also see the argument xpd.id. This argument determines where this application should look for its CAM file and actually refers to a folder in the m\adxetc\ext\xpd directory we talked about earlier. You can see that the operator application looks in the operator folder and the customer application looks in the customer folder. It's possible to define more apps which look in different folders, allowing various terminals to load many different CAM files. In fact you can

set up a hierarchy of CAM files to load using the additional arguments, not shown here, xpd.brand, xpd.role, and terminal.number.

The last file I want to talk about is adxcfgli.ccf. This file just tells which terminals to load which of the applications we just talked about in the adxcfgai.ccf file. The terminals are set in the machine range setting. You can see that the default is just terminal 10. And the applications that will get loaded are defined below.

Now that we've talked about all these files, let's switch to the controller and actually make the changes.

1.11 Editing the Config Files

Back on the controller, we need to incorporate those files we just talked about. During install, the files are copied to the adx_ipgm directory. Once there we run the ccfgtc bat file we talked about earlier. This will update our existing preload bundle and java app definitions with those defined in preload.ccf and javaapp.ccf.

Unfortunately, there is no analog to the 4690 config utility for the terminal application definitions, so we have to update those manually.

The files we need for this are in the SPGM folder.

To get started we need to copy the existing active terminal application definitions (in the file adxcfgaa.dat) into an inactive file: adxcfgai.dat. Notice the last character difference: A for active, I for inactive. Your system may not have an adxcfgaa.dat file, if that's the case, just create the adxcgfai.dat file from scratch.

If we open that file, we can see the existing applications defined for our terminals. We want to add to these, the new applications defined in the adxcfgai.ccf file that was copied to ipgm during install. Then we need to copy these over to adxcfgai.dat. If you want to define any additional composite applications, this is the time to do so. But I am just going to leave the defaults.

Now we need to repeat that process for the terminal load definitions.

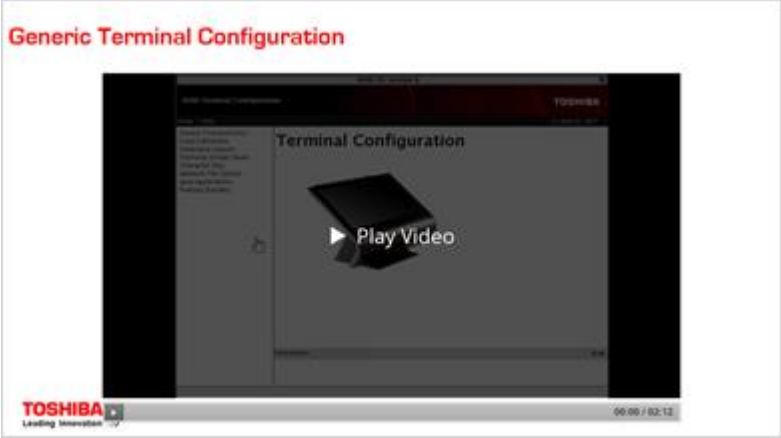
Copy adxcfgla.dat to adxcfgli.dat. If we open that, we can see the existing load definitions.

Open the adxcfgli.ccf in the ipgm directory to copy in the new definition.

This time I am going to change from the defaults slightly. You can see the machine range is set to terminal 10. I am using terminal number 10, so I can leave that alone, but my terminal does not have a customer display, so I am going to remove the line that specifies the application CCFCUST.

Now, we just made changes to the inactive files, so we need to go into the configuration menu and activate the changes. Once activation completes, we can check for errors. Since there were none, we can now go into Generic Terminal Configuration and make the last few changes necessary to get TCx Elevate running on the terminal.

1.12 Generic Terminal Config



In the generic terminal configuration menu, the first thing we want to look at is the device characteristics, to make sure that we have one set up to run with Elevate.

The first thing we need to look at is the Java redirection. We want to make sure that the advanced Java redirection is enabled and, under advanced, we don't want to use the defaults, but we do want to make sure that the I/O processor and tone and our displays are checked.

We also want to look at video displays. Make sure that the video display is enabled and that the video display is set as the system display as well. We can save these changes and go back to terminal configuration

Now we want to look at load definitions and make sure that we have a definition for any terminal that will be running TCx Elevate. In my case that's just terminal 10. Make sure that you've selected the device characteristics that we just set up. If we look at the primary application, we'll see that it's currently set to JSIFTS10.386. This is the application name for using ACE with SIGUI. You would put here the application name for whatever POS application you want to run.

Under extensions, we need to make sure that we load GTK, TDK8, Mozilla based web browser, and the two Elevate extensions: client common and client core.

If we look under preload bundles, we can see the CCFRES preload bundle that we created earlier. We want to make sure that we load that, and all the bundles we need for SIGUI to the F drive of the terminal.

Under java applications, we want to make sure that we use Java 6 / TDK8, and as the primary application select the NOOPCCF that we created earlier, and make sure this goes to the F drive as well.

Now we just need to save and activate the changes. With the changes activated, we should be able to boot our terminal with Elevate. We'll take a look at that in the next video.

1.13 Launching with CAM

When the terminal boots, it should eventually reach the message "W008 Program is Loading...". When you see this message, it's likely (if we everything was done correctly) that Elevate has launched on the terminal.

There is nothing built into the TCx Elevate launch command that automatically switched the terminal to the enhanced side manually, so, we need to do that manually by keying S1, 58, S2.

And now we can see the Welcome to TCx Elevate default page. This page doesn't do much, but it does let us know that the platform has been installed and is running on the terminal.

The Welcome page is based on a Composite Application Markup (or CAM) file that gets loaded to the terminal from the controller's M drive. Elevate uses these CAM files to define what shows up and where on the terminal's operator and customer displays.

If you are installing Elevate without the samples, then this is the only CAM file your system has on it right now, and you will need to create a new CAM to get your terminal ready for POS transactions again.

However, if you've installed the samples, we have included a few example CAM files that you can use to get a sense of how the Terminal UI is configured. So let's switch back to the controller and take a look at the M drive where CAM files are stored.

CAM files are stored in the M:/adxetc/ext/xpd directory that was mentioned earlier on in the install process. If we navigate there, you can see all of the files and folders that will get loaded to the terminal as part of the CCF resources preload bundle.

If we move to the config folder, then to the ca folder (ca standing for composite application) and check the contents, we can see the tgcswelcome.cam file. This is the CAM file that was loaded on the terminal and the file that will always get loaded on the terminal if no other cam file is found.

You can also see the operator and customer folders. These folders correspond to and argument in the launch command. If you remember, in the adxcfgai.dat file that we modified before, the one that defines the terminal applications, there was xpd.id=operator and xpd.id=customer.

These xpd.id arguments must point to a folder within the /xpd/config/ca directory. You can other arguments as well (xpd.brand, xpd.role, and terminal.number) to match a hierarchy of folders under the xpd/ca/config directory.

Now since I have installed the samples with the platform, a few sample cam files have been loaded to the operator folder. Here you can see a sample for ace, sa, and gsa. But notice that all of these files have a “.smp” extension at the end.

Elevate will always look for files ending in “.cam” in the folder specified by the arguments in the launch command, if it can't find a CAM file there, it looks in the next folder up. In this case it moved up to the ca folder and loaded the tgcswelcome cam file.

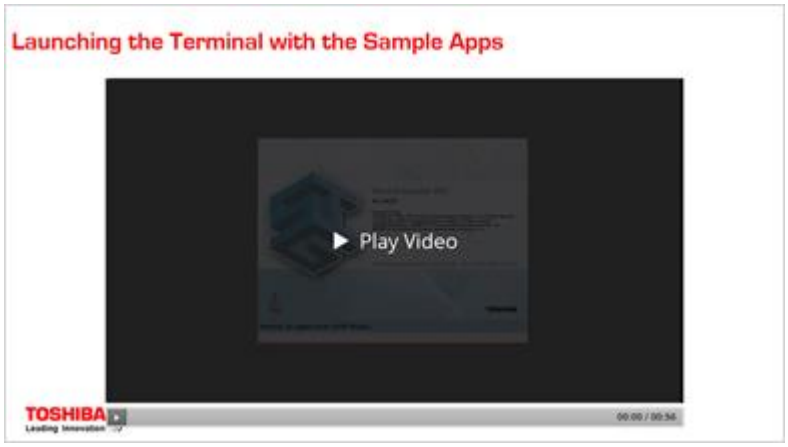
If I copy one of these samples and remove the “.smp” from the end, my terminal should now use that file at launch. I'm running ACE on my machine, so I'll copy the ace file.

Let's take a look at this file and see if we can't get an idea of what we are going to get on the terminal.

I'm not going to go into detail about the CAM language now, but even without much knowledge about it, we can get an idea, of what will show up on our terminal. Here you can see that we are loading SIGUI with the app ACE. You can see here that on a swipe down, we will raise an overlay called "Utility pane" and hide it with a swipe up. Here you can see that on a swipe left, we raise an overlay "Prescan overlay" and hide it on a swipe right. Down below, we can see those overlays defined. The utility pane overlay launches "terminal-utilities/index.html. This is the sample terminal utilities app. And here is the pre-scan sample app. So our terminal should launch ACE with SIUI, the terminal utilities app, and the pre-scan app.

One last thing we have to do is rebuild the preload bundles. Since we've added a new file, we have to run the adxpldrb command to make sure that the new file gets included in the preload and sent to the terminal. Now we should be able to reload the terminal and see our changes.

1.14 Launching with the Sample Apps



Now that we've created a CAM file in the operator directory, we can see that out terminal is bringing up SIGUI. Notice this time that we didn't have to manually switch to the enhanced side of the terminal, since SIGUI does that automatically.

Once the Terminal is up, we can log in. Right now, the terminal doesn't look any different that it would if it weren't running Elevate. Customizing the look and feel is done through the CAM files, and since this cam file specifies that the full screen is devoted to SIGUI.

However, if we swipe down from the top of the screen, it will bring up the TCx Elevate utilities app in an overlay. You can see there is an area for messages and an area for the various utility functions.

We can swipe up to close the overlay and the app.

If we swipe in from the right edge of the screen, it will open the TCx Elevate pre-scan app. And we can close it by swiping it away again.

Which apps are run on a terminal is determined by the CAM file as we've seen. The apps themselves are defined in the html folder back on the M drive of the controller.

1.15 Course Complete